1. (Currently Amended) A system for synchronizing multi-modal interactions, comprising:

a multi-modal application comprising at least a first mode process that enables user interaction with the application in a first modality and a second mode process that enables user interaction with the application in a second modality;

a multi-modal shell for managing and synchronizing information exchanges between the first and second mode processes of the multi-modal application to enable a synchronized multi-modal interaction with the application; and

an API (application program interface) that allows the first and second mode processes to register their respective active commands and corresponding actions with the multi- modal shell.

2. (Original) The system of claim 1, further comprising a registry having a registration table, managed by the multi- modal shell, that comprises a list of each of the registered commands and corresponding synchronized actions that results in both the first and second mode processes upon execution of a registered command by one of the first and second mode processes.

3. (Currently Amended) The system of claim 1, wherein the multi-modal application comprises ~~further comprising~~ at least a first mono-mode application for the first mode process and a second mono-mode application for the second mode process, wherein the multi-modal shell manages and synchronizes information exchanges between the first and second mono-mode applications.

4. (Original) The system of claim 1, further comprising at least a first device having a first user interface modality and a second device having a second user interface modality, wherein the multi-modal shell manages and synchronizes information exchanges between the first and second devices.

5. (Original) The system of claim 4, wherein the first device, second device and multi-modal shell are distributed over a network, and wherein the API is implemented using distributed APIs or protocols.

6. (Original) The system of claim 2, wherein the API comprises a mechanism for converting a mono-mode application to a multi- modal application.

7. (Original) The system of claim 6, wherein the mono-mode application is a GUI application, and wherein the mechanism provides speech enablement of the GUI application by registering the active commands of the GUI application and building a grammar for the registered commands to support the commands in a speech modality.

8. (Original) The system of claim 2, wherein the API comprises a mechanism for building a multi-modal application.

9. (Original) The system of claim 8, wherein the mechanism is used for directly programming the registry by building a registration table having user-defined commands and corresponding actions for each of the modalities of the multi-modal application.

10. (Original) The system of claim 1, further comprising an operating system, wherein the multi-modal shell executes on top of the operating system.

11. (Original) The system of claim 1, wherein the system is distributed over a network.

12. (Currently Amended) The system of claim 1, wherein the multi-modal application is a multi-modal browser comprising a first browser application and a second browser application~~, wherein the first mode process renders a first modality and the second mode process renders a second modality~~.

13. (Currently Amended) The system of claim 12, wherein the first browser is a GUI browser ~~modality is GUI~~ and the second ~~modality is~~ browser is a speech browser.

14. (Original) The system of claim 12, wherein the multi-modal shell processes a CML (conversational markup language) file to send modality- specific presentation information in the CML file to the respective browsers.

15. (Original) The system of claim 14, wherein the CML file encapsulates the modality-specific presentation information in a single modality-independent representation.

16. (Original) The system of claim 14, wherein the CML file comprises a combination of declarative markup languages.

17. (Original) The system of claim 16, wherein the CML file comprises a single file combining the declarative markup languages and synchronization elements to provide tight synchronization between the declarative markup languages.

18. (Original) The system of claim 16, wherein the CML file comprises a separate file for each of the declarative markup languages and wherein the separate files are loosely synchronized at predefined points.

19. (Currently Amended) A method for synchronizing multi-modal interactions, comprising the steps of:

activating a multi-modal application comprising at least a first mode process that enables user interaction with the application in a first modality and a second mode process that enables user interaction with the application in a second modality

receiving a command or event in ~~a~~ the first modality;

triggering (i) an action in the first modality and (ii) a corresponding action in ~~at least a~~ the second modality, based on the received command or event; and

updating application states or device states associated with the first modality and the second modality.

20. (Original) The method of claim 19, further comprising the steps of:

registering active commands associated with the first modality and active commands associated with the second modality;

associating, with each registered command of the first modality, an action on the first modality and a corresponding action on the second modality; and

associating, with each registered command of the second modality, an action on the second modality and a corresponding action on the first modality.

21. (Original) The method of claim 20, further comprising the step of building a command-to-action registration table based on the registered commands and actions.

22. (Original) The method of claim 21, wherein the registration table is built by a multi-modal shell via API calls from the applications or devices associated with the first and second modalities.

23. (Original) The method of claim 20, wherein the step of triggering comprises the steps of:

looking up the received command in the registration table; and

executing the actions associated with the received command on the first and second modalities.

24. (Original) The method of claim 20, further comprising the steps of:

registering a callback handle for each of the registered commands to notify the first and second modalities of completion of the actions corresponding to the registered commands.

25. (Original) The method of claim 24, wherein the step of updating the application states or the device states comprises the steps of executing the callback handle associated with the received command to trigger a callback action on the first modality and a callback action on the second modality.

26. (Original) The method of claim 19, wherein the step of triggering comprises the steps of:

executing first thread associated with the received command; and

triggering a corresponding second thread to initiate the corresponding action on the at least second modality.

27. (Original) The method of claim 26, wherein the threads are applets.

28. (Original) The method of claim 26, wherein the threads communicate via socket connections.

29. (Currently Amended) A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for synchronizing multi- modal interactions, the method comprising the steps of:

activating a multi-modal application comprising at least a first mode process that enables user interaction with the application in a first modality and a second mode process that enables user interaction with the application in a second modality;

receiving a command or event in ~~a~~ the first modality;

triggering (i) an action in the first modality and (ii) a corresponding action in ~~at least a~~ the second modality, based on the received command or event; and

updating application states or device states associated with the first modality and the second modality.

30. (Original) The program storage device of claim 29, further comprising instructions for performing the steps of:

registering active commands associated with the first modality and active commands associated with the second modality;

associating, with each registered command of the first modality, an action on the first modality and a corresponding action on the second modality; and

associating, with each registered command of the second modality, an action on the second modality and a corresponding action on the first modality.

31. (Original) The program storage device of claim 30, further comprising instructions for performing the step of building a command-to-action registration table based on the registered commands and actions.

32. (Original) The program storage device of claim 30, wherein the instructions for performing the step of triggering comprise instructions for performing the steps of:

looking up the received command in the registration table; and

executing the actions associated with the received command on the first and second modalities.

33. (Original) The program storage device of claim 30, further comprising instructions for performing the steps of:

registering a callback handle for each of the registered commands to notify the first and second modalities of completion of the actions corresponding to the registered commands.

34. (Original) The program storage device of claim 33, wherein the instructions for performing the step of updating the application states or the device states comprise instructions for performing the step of executing the callback handle associated with the received command to trigger a callback action on the first modality and a callback action on the second modality.

35. (Original) The program storage device of claim 29, wherein the instructions for performing the step of triggering comprise instructions for performing the steps of:

executing first thread associated with the received command; and

triggering a corresponding second thread to initiate the corresponding action on the at least second modality.

36. (Original) The program storage device of claim 35, wherein the threads are applets.

37. (Original) The program storage device of claim 35, wherein the threads communicate via socket connections.